au_to_path()

Be careful with paths passed as a parameter

Sean Barnum, Cigital, Inc. [vita[1]]

2005-10-03

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 7152 bytes

Attack Categories	• Path spoofing or confusion problem • Malicious Input
Vulnerability Categories	• Indeterminate File/Path • TOCTOU - Time of Check, Time of Use
Software Context	• File Path Management
Location	
Description	The au_to_path() function takes a pathname as an argument. Care must be exercised when accessing files from passed in pathnames. The *au_to_path(char *path) function is used to format an input path name into a path token. A path token contains access path information (token ID, a byte count of the path length, and an absolute path) for an object. au_to_path(path) is vulnerable to unknown malicious changes to the path passed as a parameter.
APIs	**FunctionName** / **Comments** au_to_path()
Method of Attack	The key issue with respect to TOCTOU vulnerabilities is that programs make assumptions about atomicity of actions. It is assumed that checking the state or identity of a targeted resource followed by an action on that resource is all one action. In reality, there is a period of time between the check and the use that allows either an attacker to intentionally or another

1. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/35-BSI.html (Barnum, Sean)

	interleaved process or thread to unintentionally change the state of the targeted resource and yield unexpected and undesired results An attacker could potentially link the passed in path to a path known by the attacker causing failure of the expected path operations as well as potentially leveraging the returned token to access files that should not be accessible at the attackers level of privilege.
Exception Criteria	If proper checking is performed or user-specified input is not used, this is not a problem.
Solutions	**Solution Applicability** / **Solution Description** / **Solution Efficacy** Generally applies to au_to_path() / The most basic advice for TOCTOU vulnerabilities is to not perform a check before the use. This does not resolve the underlying issue of the execution of a function on a resource whose state and identity cannot be assured, but it does help to limit the false sense of security given by the check. / Does not resolve the underlying vulnerability but limits the false sense of security given by the check.

When the file being altered is owned by the current user and group.	Set your effective gid and uid to that of the current user and group when executing this statement.	This will prevent an attacker from altering any file they can't already alter.
When user specification of the file to be altered is not necessary.	Do not rely on user-specified input to determine what path to format.	This will reduce exposure but will not eliminate the problem.
Generally applies to au_to_path().	Limit the interleaving of operations on files from multiple processes.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
Generally applies to au_to_path().	Limit the spread of time (cycles) between the check and use of a resource.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
Generally applies to au_to_path().	Recheck the resource after the use call to verify that the action was taken appropriately.	Checking the path permissions after the operation does not change the fact that the operation may have been

				exploited but it does allow halting of the application in an error state to help limit further damage.
Signature Details				

Examples of Incorrect Code	`#include`
	`#include`
	`/* check permissions to the path */`
	`if(!access(file, ...)`
	`{`
	`/* format path into path token */`
	`au_to_path(path)`
	`}`
	`else{`
	`/* permission was denied */`
	`}`			

Examples of Corrected Code				
Source References	• ITS4 Source Code Vulnerability Scanning Tool [2] • Viega, John & McGraw, Gary. *Building Secure Software: How to Avoid Security Problems the Right Way*. Boston, MA: Addison-Wesley Professional, 2001, ISBN: 020172152X, ch 9			
Recommended Resources	• M. Bishop and M. Dilger, "Checking for Race Conditions in File Accesses[3]," Technical Report, CSE-95-10, September 1995. • M. Bishop and M. Dilger, "Checking for Race Conditions in File Accesses[4]," Computing			

	Systems 9 (2) pp. 131-152 (Spring 1996). • Solaris 10 Reference Manual Collection[5] • SunSHIELD Basic Security Module Guide[6]	
Discriminant Set	**Operating System**: • UNIX **Languages**: • C • C++	

Cigital, Inc. Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. mailto:copyright@cigital.com